

Design Verification and Test of Digital VLSI Designs
Dr. Santosh Biswas
Dr. Jatindra Kumar Deka
Indian Institute of Technology, Guwahati

Module - 3
Logic Optimization and Synthesis
Lecture - 4
Heuristic Minimization of Two-Level Circuits

Good morning and welcome to lecture number 4 of module 3. So, in the last 3 series lecture on in module 3 what we have studied? We have seen minimization of 2 level Boolean functions or 2 for 2 level circuit. So, what we have seen that given a Boolean function, so then we how can we represent it using the minimum number of gates, that was the main idea. Because more the number of gates or the gates with higher number of inputs means the more number of resources. So, given a Boolean function in the arbitrary form, so what we have to be our main emphasis should be the main emphasis should be there, it can be represented or it can be implemented you rather say in minimum number of gates. And also the gates being used should have be, be should be should have as minimum number of inputs as possible.

So, for that we want to minimize our Boolean functions in the 2 level. So, we actually represent our function in terms of sum of product or product of some forms. So, which are nothing but our series of and gate followed by or gates or of a series of or gates followed by and gates. So, the, and or plane or and or plane kind of a thing. So, basically we see that most of the all the Boolean functions can be represented by a 2 level in 2 levels either in terms of and gates first followed by or gates or the other way round. And we also by minimization what we meant was that we have to implement, so that the number of gates required are minimum, as well as the number of inputs to the gates are minimum.

So, we have done it basically in 2 steps; first we found out the prime implicates. And then we found out a subset of the prime implicates that would cover all the mean terms of the functions. So, basically we can you will see that already we have seen in the last 3 lectures that this problem is an exponential problem or rather a doubly exponential problem in the number of inputs.

So, finding out if you are using the queens or the tabular of method of finding out the prime implicants. So, if we have to first expand your function Boolean function in terms in a, a canonical representation canonical form. So, in canonical form that is all the terms are fully expanded, now by expanding the number of mean terms can be the order of 2 to the power n in the worst case where n is the number of input. And then again we have to do some kind of compression, and then you have to eliminate out a you have to find out the compatible terms, then what you do then you have to find out the terms which are already included in some other term and you have to keep on doing it. And in the end you find out mean terms, sorry the prime implicants for that function. So, that we have already seen by the tabular method. So, in the worst case the input for the tabular method that if that is representation of the function in our canonical form they actually result in exponential number of elements in number in terms of the number of inputs.

Then we have found out that iterative consensus was another method of generating the prime applicants in which we could avoid the, what you say is this exponential problem of the tabular method. And you could get the easily get the prime implicants, but after you have found out the prime implicants then you have to find out the sub set of the prime implicants which can cover all the mean terms. Here we have seen a branch and bound algorithm and then we also we have seen that if you are not using a lower bound to compute. In the branch and bound algorithm what do you, you try to find out you we take a series of we take some of the prime implicants. And then I try to find out whether there exist another solution by including some more prime implicants and whether cost will be increasing or decreasing then I given the predefined value.

So, we what do we do we start with say, say that this is basically we represent it the whole solution space in terms of tree and each node of the tree corresponds to taking or including or excluding some of the prime implicants. And then we try to find out if there is a better solution possible along the along a path from that node. And we take that along a path means we will each path in this branch and bound tree will actually correspond to some of the prime implicants been taken and some of them being dropped. So, if you find out whether through a path whether we can still get a better solution. If it is there then you expose that path otherwise you branch and bound back and try to explore other parts. Then we saw that if you try to explore the whole tree it is going to be an exponential problem.

So, we have given some heuristics where you have the lower bound computation heuristic. So, it will actually given a node it will be a fast heuristic and it will tell you whether there exist a better solution along that path. If there exist then only you explore the path otherwise you return back and try exploring other paths. But here also the problem is problem is exponential if you forget about the heuristic path it will be an exponential problem. Because any way you have to represent all the mean terms of the function, and then you have to find out which of the mean terms will be covered by your prime implicants. So, even if you include your heuristic the problem if, if you do not include the heuristic, so there tree size will grow that is because we know that selecting a subset of primes to cover a function is in other words a subsets.

And problem is actually can be complete problem, so over all the problem is our exponential problem already heuristics is saving you some saving you by not allowing you by helping you not to traverse some part of the tree. But input space you have to have the prime implicants as well you have the mean terms. So, once you have the mean terms or any problem where you have the mean terms. Then you can directly say that the number of mean terms in exponential case I mean the worst case can be exponential in number making the whole problem an exponential problem. So, the heuristics help you to save some computation path, but actually it cannot, cannot means turn it down from an exponential problem to a polynomial problem or something like that. Why because we have to end, end list or you have to list the prime mean terms in picture.

So, all the mean terms you have to list and then you have to find out which are the prime implicants which will cover it. So, whenever you have to talk about all the mean terms in case of the branch and bound or we have to talk the mean terms when you are talking representation in canonical form is nothing but you represents the mean terms. Then also when you are finding out the prime implicants by the tabular method. So, whenever you are bringing the picture of mean terms into picture so your problem is becoming exponentially complex. So, so actually finding out the minimum what do you say minimization of 2 level circuits or 2 level binary Boolean function by the technique. we discussed in the last3 lectures that is finding out the prime implication then selecting a sub set cover using branch and bound algorithm is an actually an exponential problem.

So, the number of inputs of a function you say about 100s or 200s. When you cannot use the techniques mention in the last 3 lecture series is the in feasible of handling the

problem. So, what we are going to see in this lecture? In this lecture, we will see heuristic minimization of 2 level circuits. In this case, what we are going to do? We will be actually we now start or at no point of time we will be talking about the explicitly about the mean terms. We will have a, what do you call a Boolean function and then we will try to minimize it from that from that 1 somewhat it will be somewhat. Similar to iterative consensus kind of a thing where you have to start, with given any Boolean function. In any format, we do not we will not convert it into a canonical form rather we will try to minimize from there itself.

So, that is actually you are not expanding the terms so if you are not expanding the terms then we are being saved from the computationally exponential problem. And also we will never explicitly talk about the mean terms that is been covered. So, it will end to a minimal problem, but again it will be a totally will be heuristic. Because of no point of time will be having will have an exact algorithm so it totally in all the even from the beginning I mean in the last 3 lecture, we actually computed the prime implicates.


So, that part was an exact algorithm and we never use the heuristic to compute the prime implicates. But here there will be no question of prime implicates no question of mean terms we will be given a Boolean function. And we will try to minimize in some, some way as we will be seeing in this lecture and will be a heuristic from the day one kind of a thing. So, there will be no exact algorithm involved in any path so your algorithm will be a very fast algorithm. But again the solution obtained may be a sub optimal one or you will never get an optimum solution in the in most of the cases saw the, what do you say the solution will be a near optimal solution, that is the idea.

So, whatever I told, I discussed you that is mentioned in the lecture Boolean function minimization for 2 level implementation. The major problem is the potentially very large number of mean terms and the prime implicates so mean terms and prime implicates. So, mean terms is much larger as we know from the prime implicates, but anyway in the both of this prime implicates. And mean terms can be very high in number or, or you can say an exponentially in number making the whole problem a very difficult problem to solve.

(Refer Slide Time: 07:33)

Introduction

- To cater to this problem, heuristic methods for Boolean function minimization have been proposed which avoid computing all the primes and all the minterms in an attempt to avoid the computational cost.
- The idea is to successively modify a given initial cover of the function, until a suitable stopping criterion (in terms of number of literals) is met. The drawback is the inability to guarantee the cheapest solution, though a careful choice of the algorithms may actually provide solutions very close to the optimum (when the optimum is known) in a very reasonable time.
- In this lecture, we will discuss a simplified version of such a heuristic called ESPRESSO.



So, exact minimization procedure involves 2 basic steps finding the prime implicants and determining a minimal subset. So, actually determining the prime implicants if you use a tabular method it will be exponential. And if you are using iterative consensus at least at that part you are being saved from an exponential complexity. But when you are going to find out the minimal subset this problem is going to be an exponential problem, because you have to analyze the mean terms.

So, if you have about say 2^n terms in the worst case. So it means if the circuit has n inputs in the worst case in the order of the mean terms can be order of 2^n say. For example, everything is 1 in this case so if you have a circuit with more than 15 inputs solution space will be so high, it will be virtually impossible to do it using our computer or in other words time taken will be extremely large. So, what we will go to this problem heuristic method for Boolean function minimization has been proposed. So, which avoid computing all the primes and the mean terms to avoid the computational cost, so what is the basic idea here so that is what we will be looking in this lecture?

The basic idea is that we will give an any kind of a Boolean function we will not I mean go to find this prime implicants nor we will be finding out mean terms rather we try to minimize from there itself. And see how much we can minimize and that will be your result so obviously it is a heuristic because the time taken will be much less, but you may get a sub optimal solution. So, what is the idea the idea is to successively

modify a given initial cover function until some stopping criteria is met. The drawback is inability to guarantee the cheapest solution, but careful choice of algorithms will provide solution very closely optimal time in very optimal results in a very reasonable time. So, what did we say? So, given a initial Boolean function then what we will do? We will try to try to I mean, because that is actually a cover.

So, given a initial Boolean function it is it is representing terms of some ands and ors. So we form so there will be some product terms which will be added. So basically they are nothing but they are actually covering your function. So given any Boolean function it covers the function that is very well. Now what you have to do now again if you try to find out the mean prime implicants, then again the procedure is there as you have already seen may be an exponential case. But here we are not going to find out any mean terms or we are not going to find out any number of prime implicants rather we will try to represent the function in different, different ways. So that the number of literals total number of literals become try to become minimize and minimize and smaller and smaller.

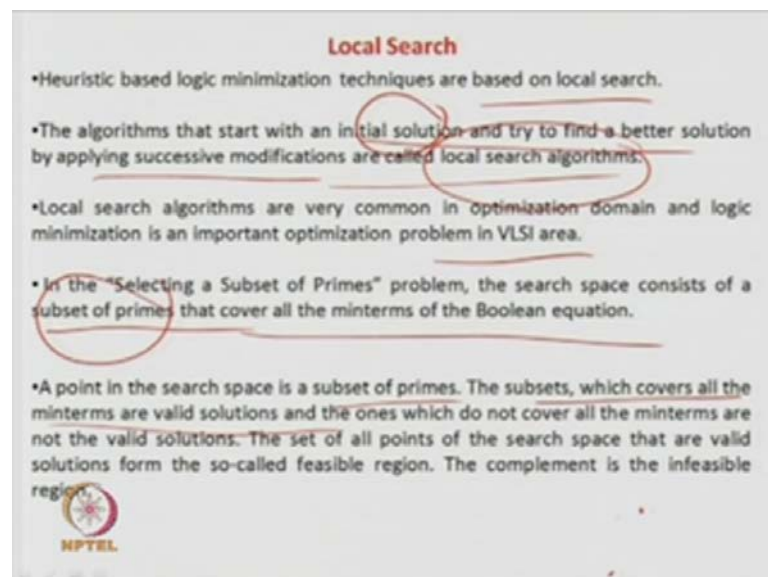
In every subsequent step we will try to minimize the product terms and the number of in terms of the product terms. So that the same function is represented, but the number of literals, start becoming less. And we keep on doing it there will be some stopping criteria so that if you have 100 literals for a function we are very satisfied stop. So 100 literals means say about we can order of some about 100 inputs and some different product terms will be there also stopping term can be some number of product terms some number of product term is that number of and gates. So what you can do you can keep on heuristically minimize the function by, by actually literal putting another. And so that the function becomes remains equivalent and what is the stopping criteria? Say if you have 100 literals or about say 10 product terms.

So that you can stop that means you require ten and gates. So and that is you have you can out some heuristic that is again the stopping criteria is also heuristic that is given a bound of the number of or the bound of the or on the bound on the number of product terms that is the number of and gates. And so forth and finally, so you keep on minimizing a function

So if you reach a stage where you find a number of literals or number of the and gates is equivalent to a stopping you stop and you produce the result. So here what is the interesting thing about this algorithm will be more the time you run these are the chances that you are going to get better and better results. Because if you set a very lower bound on the number find out the and gates or the number of literals then you have to execute the algorithm for a very large number of time or a sufficiently large number of time and then you can get a minimal representation.

But if you stopping criteria is quite relaxed like you get 300 literals or hundred 200 and gates so you can get the solution very fast, but in that case solution will not be a very optimal one. Now in this lecture the actually the name of this algorithm called is Espresso. In this lecture we will discuss a single version of this heuristic. So we will I mean till now we have been talking about exact algorithms and heuristics and what is the advantages of heuristics or the disadvantages of heuristics? So before we exactly go in to this 2 level minimization heuristic called Espresso, let us see what are this heuristics based on?

(Refer Slide Time: 12:13)



Local Search

- Heuristic based logic minimization techniques are based on local search.
- The algorithms that start with an initial solution and try to find a better solution by applying successive modifications are called local search algorithms.
- Local search algorithms are very common in optimization domain and logic minimization is an important optimization problem in VLSI area.
- In the "Selecting a Subset of Primes" problem, the search space consists of a subset of primes that cover all the minterms of the Boolean equation.
- A point in the search space is a subset of primes. The subsets, which covers all the minterms are valid solutions and the ones which do not cover all the minterms are not the valid solutions. The set of all points of the search space that are valid solutions form the so-called feasible region. The complement is the infeasible region.

NPTEL

Mainly for this logic minimization problem, so heuristic based logic minimization are based on local search. So we see what is the local search because we have till now not defined with the local search which is the global, global search in terms of this heuristics. So in this lecture well give you a quick overview of what are these heuristics are about

and what do you mean by local search. So what this algorithm will do the algorithm start with an initial solution and try to find out a better solution by successively modifying what is called local search algorithms.

So the algorithms will start with the initial solutions initial solution means the initial cover. In this case here given a sum of products from so often a Boolean function so obviously it will cover the overall the mean terms that is very well known. So that is actually your initial solution, and tries to find out a better solution by successively modifications. So in this case what you can do as we will be seeing by examples we try to modify some of the product terms.

So that sum of the products totally done you finds out some redundant terms you can be done then you can also minimize these 2 terms like. For example, if you have x as I told you x a plus x something like this, this plus x bar a is something is there then it will be a into x plus x bar that is equal to a dot 1 that is equal to a. So this type of small, small minimizations you can do, so that your product number of product terms or number of literals in the products terms will keep on decreasing.

Such type of modification minimum small, small modification you keep on doing that is actually the successive modifications and they are actually nothing but are called local search algorithms. So after doing this modification your function should remain equivalent and we should compare if the number of literals are the number of products are decreasing from the initial solution. If it is there we keep it and then again try doing further more modification this is actually called the local search.

So, local search algorithms are very common in optimization domain, and minimization in the also in the VLSI area. So local search means what we have given a solution then say you are given a solution you try to find out some solutions which are nearby. So how do you find out solutions which are near just by tweaking some of the parameters? Then if you see that whether this neighborhood solution is better than the current solution? So you take this as a present solution when you keep on repeating this.

So in the selecting in the selecting the subset of prime problem the search space consist of a subset of primes that covers all the mean terms of the Boolean function. So if you, if you consider that you are given so in this lecture we will not start with the case of the prime implicates are given to you whether we will start from a canonical I mean whether

we will start from a given function is said to minimize. But if you just take for example, the previous lecture that we have been given some prime implicants which cover a function and then you have to find out then then you have, have to say the how can you go for a local search in this? So, local search in this you randomly take a subset of prime implicants which cover this then what you do then you tweak something you remove 1 prime implicant put another prime implicant or you remove a subsets of the prime implicants from the set another subset which you still cover the function.

So, when I am saying that you remove some of prime implicants and add some of the prime implicants that is you are going from 1 solution space to another sorry 1 solution to another solution and both of them should cover your function so that is what is called the search space. So, such space is a subset of primes now what is this is one solution a subset of prime covering your function is a solution. Now, you it exclude some of the primes include some of primes and includes some of prime this is actually add the solution which is nearby you go to that solution.

And see whether it covers your function and also in the number of terms are less than the previous one if it is the case you take the new solution new solution as the current one and you keep on repeating it till you get your stopping criteria. So, a point in the suspect is the subset of primes the subset of prime which cover all the mean terms is the valid solution now this is something called a valid solution something called a invalid solution like if you take like what is the solution space solution space are nothing but subset of primes. Please remember the, this; this; this 2 points are saying are not relevant to today is lecture they are relevant to the other previous 3 lectures. Because there we had a subset of then we had prime implicants and we are trying to find out the minimum set of prime implicants which can cover the function. But today is lecture, we will not at all bringing prime implicants into consideration, but let us start working from the function in the current form. But as we are defining what is local search? So, we are trying to correlate local search a heuristic with the previous lecture.

So, what is the so if you are given in the corresponding to the last 3 lecture if you are giving a even, even the prime implicants for a function. And if we want to find out a subset of prime implicant we are not taking the branch and bound algorithm we are directly going for heuristic so in this case we can say that local search is a good heuristic. So, one in this case what you do you take a arbitrary set of prime implicants that is the

solution space a subset of prime implicates the solution space. Now, it can be valid or it can be invalid so if it is invalid means what if it is invalid that means it does not cover the function. And if it covers the function then it is a valid solution so you take a subset of prime which is the valid solution.

Now, you what you do now, you remove some of the prime implicates add some more prime implicates into the set. Now, it will become another solution now you have to find out if it is the valid solution it is fine. And if it is invalid solution you discard it if it is the valid solution you find out whether the number of mean terms products terms is less than the previous solution. If it is the case the present solution becomes your sorry this current solution becomes the new solution. And we keep on iterating it 2, till and you keep on doing it till you get a solution which has the number of mean terms and the number of product terms, the number of literals and number of the product terms as per your desire.

(Refer Slide Time: 17:49)

Local Search

- Local search relies on distance between two points in the search space.
- In case of the "Selecting a Subset of Primes" problem, the distance between two points would be the number of minterms that are covered in one subset of primes, but not in the second or vice versa (i.e., the cardinality of the symmetric difference of the two sets).
- Different problems have generally different definitions of distance, and it is also possible to define different distances for the same problem.
- Once a definition of distance is given, we can define the neighborhood of a point p as all points that are less than some distance k from p .
- Starting from an initial random valid solution, the algorithm examines its neighborhood for a feasible point whose cost is lower than the current cost. If one is found, it is assumed as the new starting point and the process is repeated until a stopping criterion is met.

NPTEL

Handwritten notes: L , 100 , 700 , 1000

So, more you do refinement the better and better solution you give and more and more time you take so this is actually called the local search kind of a thing. So, this subset which covers all the mean terms is a valid solution and the once do not cover a mean terms is not a valid solution the set of points in the search space that are valid solutions. So called the feasible region the complement is the infeasible region. So, there we defined subsets of prime implicates possible that is a power set where you the subset which actually cover the mean terms is the valid set and the others one is the invalid set.

Now, you have to find out actually you have to keep on searching within the valid set and you have to find out the solution whose cost is the minimal, we should not call it minimum which is the minimal. And finding the minimal better and if you going towards the optimality we will take more time. So, if you are giving if you are giving more time the better and better solution you are going to find the local search relies on distance between 2 points in the search space like what is the point in the search space that is the valid solution? Now, in case of selecting a prime the problem distance between 2 points is the number of mean terms that were covered by one subset of prime. But not in the second and vice versa so I will tell you what, what here what you mean by a distance?

So, in this case if you say if you forget about if you say that what are the, were the points in the solution space or subset of prime. Now, we have to define a distance, distance can be anything one, one parameter of distance can be given the number of say so this is the solution space. So, it consensus some amount of prime implicates, so you can say that the number of prime implicates or the number of literals in the prime implicates is the major of distance.

So, if there are 2 points so if it has say hundred literals and this also say 100 literals. So, we can say that distance is equal, but if you say that this has 100 literals and this has 50 literals then you can say that distance is 50. So, you can also have a complex distance function like you can say the number of prime implicates that is number of product terms in this case and the number of literals if you add and or make a formula that is the major of a distance. So, if you 100 literals and say 20 prime implicates and this is say that 50 implicates and 50 literals and say around 15 prime implicates. So, this will correspond to sixty 5 and this is corresponding 120, so you can find out the distance.

So, in this case this is actually this is the, this is the distance whether if you are talking only about the valid solution points. But if you are talking about the general case so you can have a distance whether you have a very similar way you can say that in subset of prime problem distance between 2 points can be the number of min term that is covered by a subset of primes. So, actually here in the second point it is stating that we are considering at all the valid points. Now some of the solution space that is subset of primes may not cover all the mean terms and some of them may cover all the mean terms. So, I mean distance measure can be the number of mean terms that is being

covered by the prime implicants. So, if here actually that is that is actually cardinality of the symmetric difference of the 2 sets.

So, there can be various distance measures so if you are considering all points in the number of mean terms covered by the subset of prime implicants can be a distance measure or now if you are considering only about the valid solution. So, you can say that the number of prime implicants in the set plus in number of literals in the set becomes a distance so difference between 2 points can be easily find out. So, different problems may have different definitions of distance and also it is possible to define different distances for the same problem. So, that is what I told you so number of literals can be 1 distance measure number of literals plus number of prime implicants can be remain measure or also only the prime implicants can be measure. And so forth once a distance definition of distance is given, we can find out the neighborhood of a point p as the points which are some less distance case from this.

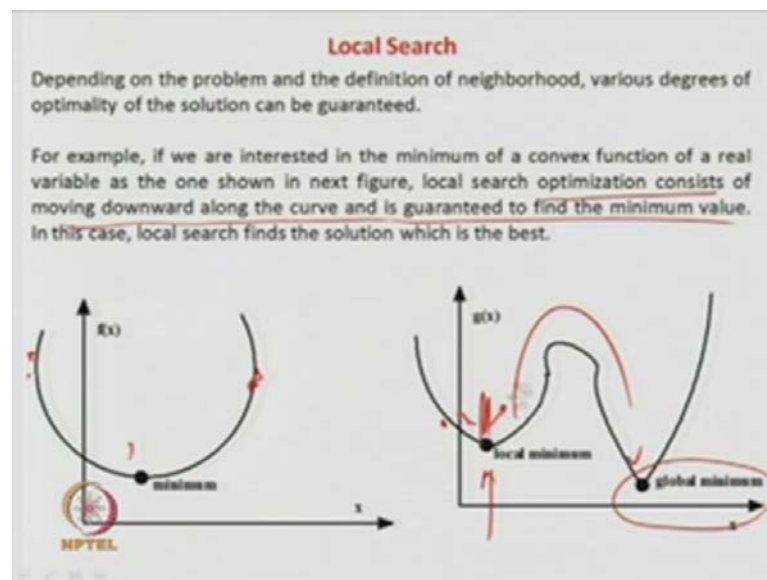
That is what our ideas is in local search what we do, we first have a solution then it would not go to a solution which is very far what we do is that we try to find out a periphery of the region whose distance is less than. Then we try to find out we try to find out the nearby solutions to this point and we find out if there is any solution which is better than the present solution that we make as that new, new solution as the present solution. And then we again try to make a neighboring part of it and try to find out which is the best solution among this and we keep on doing this.

So, if you even understand that if this is a very good solution sometimes you may reach up to these points. So, why we actually go for a local search because if we I mean too much change the distance then you may not lead to a, you may not settle down to a good solution you may always be oscillating you may get a good solution then you get a bad solution. And then again you get a good solution and so forth so that is what we do we actually increase our movement of the points in the solution space or the search in a solution space in a very slow manner that is why it is a local search we first search take a point search in a nearby spaces.

And find out if there is a better solution there and then you take it then again slowly move not that it start from 1 point and quickly jump to another point and then again see, what is the case and again come back. So, you may not converging very you may not solution may

not converge you may keep on oscillating. So, we start from our initial random solution the algorithm examines the neighborhood for a feasible point whose cost is less than the current point. And I told you if it is found it is assumed that the new start point and the process is repeated.

(Refer Slide Time: 22:11)



Till you get what you say you still you get a I mean the better kind of solution now you are representing in a graph. So, let us let this be a function in terms of f of x so this is your actually I mean this is this is your function and in terms of a graph. So, you can also arbitrarily you can start off with this point. Now, what we will do? Now, we can find out the solution space nearby neighborhood so you will you can find out that this is the base solution in the nearby point. So, this will be a next solution.

Now, what you do? So, these are the random start point then you find out that this was the next better solution the random point. Then again according to this neighborhood you find out next solution is solution one; this solution 2 then you find out the next best solution. Then you again put a boundary along with that is your again next neighborhood and then we can find out this is the best solution you keep on doing. And then you will find out that I mean this bottom then wherever you go here or wherever you go.

So finally, we will reach here then whether you go this side or go this side the solution will not be a good solution so you can say that is my stopping criteria and you can find out the minimal one. So, in this case if the for example, this is actually if you observe there is a

convex function. So for example, if you are inserting the minimum of a convex function in the real variable as one shown in this figure local search optimization consist of moving downward along the curve and guarantee to find the solution with the minimum value. So, if your function which you are trying to minimize or you want to find the optimal point in it if it is the convex part like this. Then obviously, local search or local heuristic is going to give you the optimum solution that is guaranteed. Because first do this then you find out the nearby then you here then you go here and you have to keep, but you have to keep on moving it same amount of time.

So, if you say that my stopping criterion is very good I mean very, very relaxed. Then you can stop over here you can you can find out that at this point my, my I could have get the got a better solution. But I will stop here because my time is critical and I do not want to go about for more amount of time. And this is also a good solution by my terms of I mean criteria so you can stop over here. But given a sufficiently amount of good time the local search will actually bring to the minimum point, but life is not always very simple. Because general a solution may not be as convex I mean not be a convex like this the, the solution space can be something like this.

So, in this case what can happen that your solution may get stuck in something called a local minimum that you say you start with this point then you keep on you then you make a point you come here. Then you make a point you come here then you come a point you come here then you come here. And then you find out that if I go this side I mean value will increase. if I go this side my value will increase so this is it will appear that this is the best solution for, but actually this is not. Because after rising then actually we may after raising this was the best solution, but if you can somehow throw out of this best solution for the time being then you can jump over this. And then you can find the global minimum this was a local minimum. So, local such as actually will stuck in this type of local minimum point if your function is not a convex function. So, in case of this 2 level Boolean minimize channel are so they are neighbor as smooth function as this.

So therefore, our heuristics always gets talking something on this local minimum and generally we are not able to find out the global minimum which is the optimal solution almost optimum solution. So, we guess talking the local minimal so these are the optimal solutions, but there are some techniques which will not discuss in the last lecture. That is

sometimes what we do is that if you get a local minimum and you find out this is not I mean this is both way increasing.

But still you have some computation time left as per your criteria. So, we can give a jerk to this 1 jerk to this 1 so that it can over run this 1 and you can try to find out the global minimum. But that is I mean very I mean advance stuff advance stuff we are not discussing in this. Now, for the time being this has to understand that why search heuristics do not give a very good result may not give an optimal result, because you may, but if you start with this point at the first starting point. Then obviously, we will get the global minimum which is the optimal solution, but if you start from here then you may get the local minimum.

(Refer Slide Time: 25:49)


Local Search

However, if the function is not convex, then local search may get stuck in a local minimum.

Once the local search reaches the local optimum point it terminates because it cannot see a better solution in the neighborhood. However, if it takes a worse solution compared to the local optimum then there is a chance that we may reach the global optimum.

In the first case, we say that the solution is **minimum**, whereas in the second case we say that the solution is **minimal**.

In general, one interesting property of the solution of a local search algorithm is local optimality. A solution is locally optimal if its neighborhood does not contain any solution of lower cost. In order to guarantee local optimality, it is sufficient to use the stopping criterion--when there is no cheaper solution in the neighborhood of the current solution then terminate.

 NPTEL

So, that may also happen, therefore we sometimes may or may not get the most optimum solution. So, if the function is not convex then the search may get stuck into a local minimal. So, at this point you may not get the best solution possible once local search reaches the local minimum point it terminates. Because it cannot see a better solution in the neighborhood however it takes the worse I mean however if you takes a worse solution compare to the local minimum there is a chance that we the global optimum. However, if you takes a worse solution compare to the local optimum that means what that is you have to give a jump that is if you take the take a worse solution than the present one that is you

are giving me the throw for some time then you can cross this hump and you can reach the global minimum.

But again I mean you have to again decide at what point you will go to this we can I mean take the worse solution compared to the good solution and so you can try, try the hump because is not very simple. So, if I, I do not know whether I mean local minimum or I do not know whether any global minimum. So for example, if I am here then I find out this is also a bad solution this point will be a bad solution. But if you want to accept a bad solution or a worst solution, that the present one to, try to cross the hump then you can try to move the solution point. Then you will land up here and you will find out that this was the better solution than this one so your own effort and time has been lost.

So therefore, I mean if you if you get the minimal solution still you are trying to cross the hump by selecting a worse solution if may not always be a fruitful result. So, these are very difficult decision to make that when you have to take that local minimum or when you have to forgot about the local minimum take a worse solution and try jumping about it. So, in the first case we say that the solution is minimum or optimum and in the second case we call this solution is minimal, so this minimal or local optimal or something like that. So, in most of the heuristics in case of circuits, we get the optimal solution or the optimum solution. So one general interesting property of the solution is a local search algorithm is local optimality a solution is local optimal if its neighborhood does not contain a solution of a lower cost.

In order to guarantee local optimal it is sufficient to use the stopping criteria where no cheaper solution exist in the neighborhood. So, we can also think about the different type of stopping criteria as I told you. So, one stopping criteria is that you define a neighborhood and if there is no better solution in the neighborhood then also we can stop. So, that can be some of the idea, but for the, for the heuristic we will be generally using we generally type for the local search only. Because searching for the global optimum may be a very difficult and time consuming.

But sometimes you also try that because if you have more computation time is given then we can try of even crossing the humps. So, we will just this was some of the basic ideas of what you call the heuristics local search local optimum global optimum. Because in the first design part of this course I mean in the last 10 or level lectures, we have always

discussing that this is an exact algorithms and heuristic heuristics may give you a optimal solutions one of the optimum solution. So, why all these happens?



(Refer Slide Time: 28:36)

Local Search applied to Logic Minimization

- Local search based logic optimization starts with a random subset of the primes, which is a cover. The cost of a solution is the number of cubes.
- If two solutions have the same number of cubes then number of literals can be used as a secondary cost.
- The neighborhood of a solution is the subset of primes that are obtained from the solution by adding (or removing) exactly one literal to (or from) one of the cubes. A new cover thus obtained is a valid solution in the search space (and considered) if it represents the same function (i.e., covers the same minterms).
- The solutions with more cubes than the current cover, are not considered.

Boolean function $f(x,y,z) = xyz + \bar{x}yz + x\bar{y}z + x\bar{y}\bar{z}$ and the dotted lines represent the initial cover. The cubical representation of the function is as follows

xyz	$f(x,y,z)$
000	1
01-	1
-11	1

So today, we give you a just a overview, now we will see. How local search is applied to logic minimization then we will go for heuristics express. So, local search based heuristic logic local search logic optimization what is random subsets of primes. So, please do not confuse this with Espresso, Espresso algorithm. In this case, we not start with the prime implicate that will rather start with the general given sop formula and keep on minimization using a local search. But this local search example where giving on the example where prime implicates are there that is previous3 lecture series lectures which we have done. So, the, we are applying the local search to that results. We are not applying it to express or the heuristics will be trying to which will be looking today for what you say this logic minimization.

So, logic minimization heuristics if you talk about Espresso, he does not talk care take care anything about the prime implicates, it will take a directly s o p form and start minimizing, but this local search we are applying the logic minimization of the previous lectures3 lectures with prime implicates available. So, initial start with the random subset of primes which is the cover. So, what you can do? You can take a random subset of primes and find out if it is the valid solution. So, if this is the valid solution you should cover the function and if it does not cover the function. Then you can say invalid solution

and try for another random subset the cost is the number of cubes. So, as I told you cost distance whatever you can say in this case is the number of cubes it can also be the number of prime implicants it can also be the number of literals anything can be there.

So, if the solution that the same number of cubes then the number of literals can be used as a secondary cost as I told you neighbor neighborhood of the solution is the subset of primes that are obtained or to the solution by adding or exactly one literal to and from one of the cubes I will tell you what does that mean. So, that means if you have some number like $x y z$ prime so plus something, something is there. So, what is another point and in which is the how you find out another alternative solution? So, in this case you see that I remove this, the neighborhood is found out remove this neighborhood of a solution is a observe they are obtained form, the solution by adding or removing exactly one literal from the one of the cubes.

So, you take one of the cubes and use at least one of the literals and new cover is obtained you have valid solution and consider if it represent the same function. So, if you just remove this need not become a invalid function also, if it becomes a invalid function then you have to throw with it. But if it remains a invalid solution then you can I mean whether the cost cast has reduced, because you have to remove one literal. And then you can think that this can that is what we are minimizing it. We start with the given set prime implicants then what you do so the cost here is number of literals or the number of cubes you can take now what we do we have to minimize. So, what you do you have to minimize? So, what you do you remove one literal from one of the cubes? So, that is the next solution you can think so we are slowly moving in the solution space.

So, as I told you another way of finding out the solution could have been another n numbers subset of prime implicants which is valid and you keep on doing it till you get the minimal number of prime implicants. That can also be one of the solution one way one way of being the local search. Now, what we are doing here? We are trying slightly different. So, in this case, we take a random subset of prime implicants then we take one cube and remove at least and one literal from this one. So, while removing one literal means we are actually trying to get a minimum number of solution. So, when you are randomly selecting the prime implicants so your criteria should be so initially. So, you have taken a subset which is having 10 prime implicants.

So, now when you are selecting another set of prime implicants; obviously, the number of literals or the number of prime implicants will be lower than the one in the current solution if you are taking that approach. But in this slides which you are presenting, we are taking a slightly different approach. In this case, we are not taking another subset of n number random subset of prime implicate the solution later what you are trying to do from one of the cube we are one literal. So, what it does you will actually modify the function. So, if you just take whether you still covers the function or notify it still covers the function and is equivalent then it is fine. And if and if it remains equivalent and still cover the function that means you are removed one literal. And you have reduced the cost; you have to keep on doing it till you can you are satisfied to the, the cost of the function.

So, then new cover is a subsets is the, is the valid solution then only it has to be considered and it represent the same function. So, otherwise you have to again replace back this literal and try it some other solution means solutions is more number of cubes. Then the current solution are not considered so that means obviously, no value it will be any point. So, I mean whatever I told more easily illustrated if I give you an example so let us take this function so $x y z$ prime this one and this one.

So, you can find out that they are nothing represent the initial cover the cubic representation is as follows. So, we can see that this is these are all 0. So, it is represented over here, now if you take this 2 so you can see that it is nothing but $x y$ prime sorry it is x prime y . So, it is x prime $y z$ and z prime so if you merge these 2 we are going to get $0 1$ dash. And then again if you merge these 2 so if it is nothing but $y z$ so it is $1 1$ and it is x and x prime. So, it is this way, so you can represents the this function in this way very easily $0, 0, 0$ the answer is $1 0 1$ dash is the answer is 1 and $1 1$ the answer is 1 .

(Refer Slide Time: 33:39)

Local Search applied to Logic Minimization

Boolean function $f(x,y,z) = xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z}$ and the dotted lines represent the initial cover. The cubical representation of the function is as follows

xyz	$f(x,y,z)$
000	1
01-	1
-11	1

NPTEL

So, now we can represent in a cubic fashion so what is the cubic fashion the cubic fashion is simple representing a cube like this. So, this point we will represent all 0 s. So, this direction is z; this direction is x and this direction is y now, so this point is obviously, 0 0 0. So, if you move in this direction it is x is becoming 1 y and z are still 0 if you move in the top direction the z point with the 1 others will be 0 if you move in this direction obviously, y point is become and 0.

So, what will be this point? In this point, you can see that have moved top. So, we have moved top means z is becoming 1, we have moved in this direction so that means y has become 1. But still you are in this form you have not gone in this form so x is 0. So, in this way, you can make out this whole cube can be explain. And obviously, in this point where everything is 11 so the answer is 111 and it is the terminal one.

Now, you see we represent the point 0, 0, 0; so this is the initial so it is 0, 0, 0. Now, it is 0 1 1 dash that means what it is 0 1 1 and 0 1 0, so 0 1 1 is this point and 0 1 0 is this point. So, obviously if I make this, this is this covered corresponds to 0 1 dash and again obviously. Now, you see that it is 1 1 and 0 1 1, this term so this is nothing but 0 1 1 and 1 0 1. So, these term 2 term if you combine that is this 1; this 1 is nothing but 1; this 1 is your this thing so you can find out that. So, you can easily see that this is what is going to be your, what is going to look like something like this.

Now, I will tell you what does it mean, now you can see that this is your initial cover, now what the idea was that, we have to try to remove some of the literals, from some of the function and see still it works or not. So, if you readout this function it is called x prime y prime z prime, if you just look at it, now, we can easily see that let me try to remove some function, some literal from this one.

So, now if I remove the literal this y prime remove from this one, so then what is this going to become? It will become x prime nothing will be z prime that is 0 dash 0 , so what does it corresponds to, if you correspond to this 2 terms. So, now if I this was, the this one was 000 , so if I eliminate this one, so it will become something like this, so obviously by eliminating y prime from the first term makes the valid solution, because it will still cover the function and no more new in terms of enclosed into this one. So, still remains a valid solution, and then you can find out that this second term, that this one 0 1 dash 0 1 dash will become a hidden and cover, and it can be removed.

So, your function will become 1 , it will become x prime dash y prime plus the last term will be dash y z this one will be your final function, because this middle term you can do now. We see so how can find out y is the valid solution because by during y you can find out that I mean this point will actually grow into this one, this was initial point. So, it is a valid solution, because all the mean terms are covered, and no extra term is covered. But say if I remove x prime, so if I remove x prime from this one, what it will become, it will become dash 0 0 , that means it is 0 0 0 and it is becoming 1 0 0 . So, this is actually covering this 2, so this is actually dash 0 0 , so that is, if not does not at all remains a valid solution, because it will make a cover like this. So, it is an extra mean term which is covered, which is now actually not allowed in this case.

So, this will so removing x in this case, will not give you the valid solution, so only the valid solution will be x prime y prime z prime, so y if you remove, then it will remain the valid solution. And then this will becoming your this one and your, so your terms become a better solution. So, what we have done, so our initial solution was this one, you can find out that, this may be this set of prime implicants. So, here in this case you started with the set of prime implicants, it may also not start with the set of prime implicants, you can directly start off with this function set also, that is also possible. So, in this case you find out there are 3 primes, you started off with this one, then you actually start eliminating one literal from any of the terms.


So, in this case I started you start with anything, you can start with the x prime and from this one, it does not became a valid solution, it because it becomes this way. Then you start eliminating y prime, you make eliminate y prime, then a solution will be nothing but it will look like this. And then you can find out that this becomes a redundant stuff and it can be removed. So, it is the better solution, than the previous one and you keep on doing, it till you get your desired solution or what or you can fix a stopping criteria like, if you think that in this case the solution is 1 2 3 4 4 literals are there 2 terms are there. So, if you think that 4 literals and 2 terms are there is very fine for you can stop here, otherwise you can start eliminating z from something like this is you start eliminating x x prime for this. And see, what happens and keep on doing it and you have to do it.

(Refer Slide Time: 38:25)

Local Search applied to Logic Minimization

- Now let us remove the literal "y" from cube $\bar{x}yz$ leading to $\bar{x}-z$.
- It may be noted that we can do so because $\bar{x}-z$ comprises minterms $\bar{x}yz$ and $\bar{x}y\bar{z}$, which are to be covered in the function under question.
- After removing the literal "y" from cube we get the cover as shown in last figure (c). Now we can remove the redundant cover, i.e., remove the output literal of $\bar{x}y-$, thus effectively removing the cube from the cover, we have the cover as follows.
- Here, cover is the optimum solution.

xyz	$f(x,y,z)$
0-0	1
-11	1


NPTEL

Till the solution should be better as well as, and in the same case, we should be having a valid solution, what you mean by valid solution that you should cover exactly the same mean terms which was been covered by the initial solution. So, let us now, since what I told you you remove y from this one, which is leading to this one, it may be note that, we can do so because it covers mean term this, and this which are to be covered in the function under question. And it is not x any way covering any additional mean terms, after y from cube, we get the solution in figure c, that is what we have shown. And then so we can remove this redundant cover, and this is your actually optimum solution. In this case is optimum but in general speaking case you have to keep on doing this, till you get the desired solution on your time is over.

(Refer Slide Time: 39:08)

Local Search applied to Logic Minimization


The function is ω as follows

$$f_1(xyz) = \overline{x}yz + x\overline{y}z + xy\overline{z} + xyz; \quad f_2(xyz) = \overline{x}y\overline{z} + x\overline{y}\overline{z} + x\overline{y}z + \overline{x}yz + xyz$$

The cubical representation is

xyz	f_1	f_2
0-1	10	
1-0	10	
00-	01	
-00	01	
-11	01	

The initial cover of the two outputs of the function is illustrated in next Figure.



So, now we will see the same thing for a 2 way 2 output function same thing will happen 2 or 2 output function, so I mean you can think that this is f_1 and this is f_2 . So, these are the 2 functions the side we are representing this, so these are these 2 terms are representing the first term f_1 , so f_1 is 11 other f_2 is 0. And now, for this 3, we are actually having the second output as 1, because this, represent f_2 and this representing f_1 . So, now you see, if we take these 2 x prime y prime z , and x prime y z , so actually this can be presented in terms of x prime x prime means is 0, y y prime cancels out and it is 1 0 1, so this is this 2 represents this actually this one.

Now, you can see if I take this 2 this x y prime z prime, and then x y z prime, so in this case you can eliminate this 2, so it will be 1 dash 0, so this 2, this 2 terms represent by this one and both these terms are 1. Similarly, if you take this 2, so if you take this 2, it will be generating this guy now, if you are taking the last 2, we will be generating this guy. And I think if you take x prime y prime z and x point prime z prime, so in this case dash 00, if you take this 2, if you take this middle 2 guys, we will generate this term, so middle 2 terms means x prime y prime z 1.

So, the 0 0 is generated by this 2, 0 0 generated by this 2 term, so first 2 terms, this 2 terms are generating 1 this is 1; this 2 terms that is this term, and this term is generating term number 2. Because x prime y prime z prime and x y z prime x has cancel out, this one and the last 2 terms, these 2 terms are generating the term number 3. So, in, in all

these cases, you can see in terms 1 2 3, this last 3; this; this; this is this, these 3 corresponds to the f 2, and this corresponds to f 1. Now, this is now you can represent it in this way.

(Refer Slide Time: 41:19)

Local Search applied to Logic Minimization

xyz	$f_1 f_2$
0-1	11
1-0	10
00-	01
00	01
-11	01
0-1	

It may be noted that there is no way to improve the solution by expanding the input parts of the cubes or reducing their output parts. However, if we expand the output part of the first cube (i.e., 0-1|10), we obtain the cover above.

NPTEL

So, I mean I need not go, go about and how do you represent this, so in this case you can easily find out that 0 dash 1 is nothing but it is you can say that, this will represent this one, and this one is representing 1 dash 0, you can think is represented by this one. So, this is how you represent the cover, so this is your cover for f 2, and this is cover for f 3. So, in this case you can think, how many are there, this one 2. So, this one 2 are here and in this case it is 1 2 3 are there so but 4 are there, we will tell you, how it happens. Now, in this case you see, so here you have interesting thing you have to find out, that you can go about minimization for this one. So, in the previous case what you have done, so we are trying to expand, I mean we are trying to I mean take out some of the terms from this one.

And, we are trying to see, if we can get some random covers, if you are trying to take out some literals from this one. And then we find out which still remains a valid solution and the cost is decreasing, and so forth. So, in this case, if you see, so you, you will really find that, we cannot do much over here. And in this case, if you look at it, so what are the solutions we have to find out?

So, we will see what happens here, so in this case this; this; this value corresponds this one corresponds to 0 0. Next, let us see what this; this each of this things corresponds to, so this is x, so x is going both the way, in this so we can think that, it is dash 0 0, so this cube, this cube corresponds to dash 0 0, because x is moving from here to here, so this one is corresponding, if you consider 2, so this is actually this number 2. Now, you can see this is dash 0 0, 0 0 dash that means, in this case z is moving, both the directions which direction is z, so z we can find this top direction, so in this case z is moving both the ways but x y are 0.

So, this one actually corresponds to 0 0 dash, so this corresponds to 1, so you see 1, so this is actually the 1 because this is 0 x is 0 y is 0, but z is moving both the direction, so this is this corresponds to num 1. And then we have term 3 term 3 is 11, and it is actually x is moving both the sides, so in this case you can think, that this; this I mean actually x is moving both the sides. But other terms are y and z are 11, so this is dash 11, so this is nothing but term number 3. So, we will see, how this has come, we will find out; we will tell you later, how it has come up in this case. So, this; this; this; this, this one corresponds to this term dash 00 this term 2, this one; this; this one; this one actually corresponds to 1, that is this one, and this term is corresponds to 3. In this case, but how this; this row comes? We will tell you later, so assume that this; this; this, the question mark eclipse is not there, then how do you minimize this?

So, in this case whatever I do you can find out that there is no minimization possible, because if I, if I remove this term something like this it will become this one, and we cannot there is no way of merging it, because in this case, if you look, look here. So, we are having a very good way of merging it, because I mean here, if this; this was there; this was there just, we had something to include this, so we could have eliminated this. So, in a very singular philosophy you can just think over this, just assume that this question mark is not there, and if you just look at this there cannot be any minimization possible. So, that means, if I mean, you can make something like this, you can make soothing like this, but you cannot remove something, and make a correlation between this point, and this point diagonal connections are not there.

Then how do you tell say about this say for example, 000 and this is a 001. So, how, how can you make a cluster between these 2, because 0 0 and this is the only difference 0 0 1? So, this is 001 and this is 00; this point so 1 bit is different. So, you can actually make a

cluster in between this, that means you can eliminate 1 literally if you eliminate, then this point will become this, like this. If you eliminate some term, some literal term from this one, it will make, but you cannot have a merger in between the diagonal part, because merger in between the diagonal part, means you have to eliminate may be 2 or more literals, it is just like a consensus. In case of a consensus, if you remember the tabular approach, so we compare sometimes like $x y z$ and $x y z$ prime. So, we just compared in between these 2, because there is 1 bit difference over here.

Similarly, this is a very analogue in this picture, so that is, if you try to eliminate something, so it will start making clusters. In this way or in this way but you cannot have a cluster in diagonally means clusters in diagonal means, that means there are 2 terms, which is different like in this case it is 000 and in this case say it is x is 1 over here, z is y is 0 over, x is 0 over here y is 0 over here z is 1 over here. So, we can say that 2 bit difference between this 2 points, so you cannot have a something like this, so there is no minimization possible in this case. Similarly, if you just eliminate, if I do not think this one is there, so I mean if you do not, if this one does not exist, so same philosophy will hold.

Now, what this people have done? This people have found out that these are, these are different way of minimizing this one. So, what they have done? They have found out that these 2 terms are actually included, in this say if we call it A , and this is call it B . So, both of them are also available in function number 2, and but the 3, I mean terms which is considering in this 3.1 is this one 2 is this one, and 3 is this one. So, this, this middle, I mean this guy is not presenting f_2 , but it is present in 1. So, what they have done, they said that so initially, if you consider f_1 , so this one will be this and if you and and this term is also not available over here. So, if you just have a look at it, this is not there but this one 01 dash 01, if you make it 1, and add it over here, that is what is being done over here, slightly different philosophy is there 0 dash 1. And we can also think that is 1, and it is still be still a valid solution.

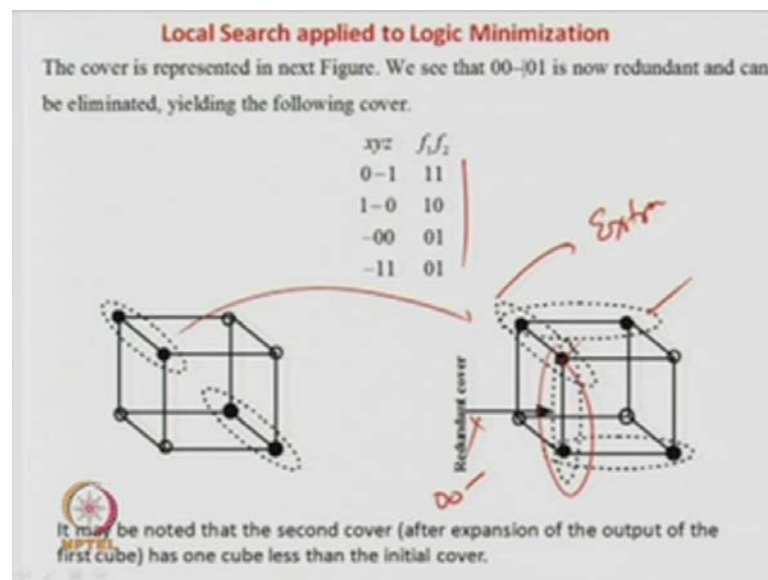
Now, we have valid solution because you see, this point is included over here, and this point is included in the other, and so we can easily have it over here. So, they can say that, if and both of them are giving $A=1$ in both the cases, in both f_1 and f_2 . So, they are just look this guy from here to here, and they have added instead of having 10, if you

remember in this they made it 11. In other words, what they have done they have made 0 dash 1, 01 and in this case, they have and in this case, it was A this one if you remember.

Now, they have just made a club of it, so in this case you make a club of it, you are going to get a something like this, so this is some different way of finding a different solution space, we are not exactly discussing the algorithm how it is done. Because, say when it was a single output, it was very simple in which case you have to find out the nearby solutions. And then we find which is the better 1, and so 2 output functions, then you have to combine these 2, and see what better you can find out.

So, in this case you can see, that this is the part which is common, because for this 2 points f 1 is also 1, and this 2 points f 2 is also 1, so you take this one, you borrow it from here to here, that is what has been done. So, that is what is been written that is there is no way, the solution can be improve by expanding the inputs or the outputs. But if you explain the output of the first cube this one, we obtain the cover as above this is what we have done, so here exported it from here to here.

(Refer Slide Time: 48:37)



And now, we get go about for the solution, now what they have done, so they have you can find out that if I put this one as extra, we have borrowed it from here to here. Now, we can find out that this one is redundant cover, because this one is taking care of this point, this guy is taking care of this point. So, obviously; this, this part will become a redundant part, and it can be thrown off, so this was actually considering. So, this was

considering point 1, so that actually 00 dash, so this is 00 dash actually it was considering this cone, which can now be removed. Now, it is represented in terms of 4 terms 1 2 3 4, since where reduction has been possible.

So, what I am, what I am saying is that, so what I just want to emphasize is that, so both the number of outputs or whenever you move from single output to a multiple output cases, the searching the, neighboring solutions, become a more sophisticated procedure. In the previous case single cover, what was the next solution? The next solution was just eliminating 1 of the literal, and you could find the neighboring solution, and keep on doing it. But if you have multiple output function the things are not very simple, so this is 1 function; this is 2 function; this is 2 output function. So, now A 2 output function; obviously, the circuit is going to, if a circuit has 2 outputs some logic gates are shared by both the outputs, so that, that philosophy has to be brought into picture. So, finding a neighboring solution here does not, may not resolve to just like removing 1 literal function, finding out a solution.

So, if you look, if you just think about this, because in this case; this will not be there, so I think this was your some 1 2 3 is there, so this are these things only correspond to some of your terms, so this is dash 11; this is dash 00 something like this; this one is not there. So, all these things are there, but now we can think that for these 2, and these 2 both of the function are 1, so you borrowed this guy from here to here. And then you can say that this one becomes a redundant cover and remove it and you can get a better solution. But there is no elimination over here, this more or less remains same where, there is no optimization, only in terms of 2 there is an optimization. So, for finding the neighbouring solution in this case, it was not just about removing 1 literal in doing, but rather doing some export and import from each other. So, that is what is being done, so it is been move it from here and here. And then you can remove this as the cover, here it remain same as the old solution, that is these 2 will be there, but we could have said something, in the f 2 business.

So, what apart I am saying that we are not exactly telling, how we have got this, in case of single output, the answer was remove some literal. And see what is happening, and what is not happening, but in the case of multiple output, you have to take some solution from point function a to function 2 and solve the body. So, therefore, what I am emphasizing is that, in case the heuristics by local search is very simple take a random

solution finding a neighboring solution, see if the cost is good, keep it else reject it and keep on doing it. But moving from solution space 1 to solution space 2 is a very, very difficult problem, because heuristic otherwise heuristic design you have been a layman's job that is you take a solution, take another random solution, if it is good keep it else remove it, and keep on doing it. But finding the next better solution is way how your algorithm time will be considered.

So, if you take n number, so if you say that I want to take a next random solution. And the way your next random solution will be taken is totally arbitrary, and you do not pay any attention to that algorithm, which is finding out the next random solution. So, if you keep on try this, try just searching in the random solution, without any hint, and then you may try to land up, you will try to you may take a very long time to take a good solution. But if your algorithm is good, then from the current solution to the next solution, you go, you think a little bit. And then you go to next solution, like in the case of multiple output function, which was not as simple as moving a literal, it is something you think about by merging something and so forth, and importing some of the terms from output 1 to output 2 to get the next solution. That is the next solution, you are selecting intelligently then your search will be quick and quickly, you may go to a optimal solution.

This is very similar like, say you are in a forest, and you are trying to track some animal to shoot down, like for example, in case of now if it is a I mean hunter or either intelligent person. Then what you will do? You will try to hear the steps of the animal, it is targeting then you try to look at the, what do you the call foot prints of that. And then you try to find his way, not directly does not know the point where the animal is, he is also trying to look for the animal, in a heuristics manner just but by looking his experience. He is trying to find out which way he should go, if you find out the footstep footing print of the animal then you will try to go in this way.


But the person who is totally un-experienced or un-experience hunter, so what he will do? He will will first try to run ten 10, 10 meters may be see in 1 direction. Then you may try to run in another n number in 10 meters 10 meters, so for in finding the animal will be a very long process. But for the intelligent person who is just taking some hints, that is to find out is next course of action, he is paying lot of intelligence, so he can get to the optimal solution, very quickly. That is about heuristics also current solution maybe random but finding out the next solution, you are going to try out is a very intelligent

problem, that you have to find out, how all the neighboring which 1 you should target next, and see its quality, so this is how it goes about.

(Refer Slide Time: 53:28)

A Simple Local Search Algorithm for Logic Minimization

- The examples in the last section suggest that a procedure based only on the simple moves that immediately decrease the cost is effective.
- However, in the real scenario we need to consider a wider variety of moves, and combine them into sequences of moves that eventually lead to the desired reductions in cost. In other words, we may also need to consider moves that may not immediately reduce the cost.
- Now we discuss an algorithm which applies simple local search algorithm for logic minimization.



So, now so I mean, so in the example last section, I mean last class, we suggested a procedure based only on the simple moves that immediately the cost decrease is effective. So, what it is saying in the examples, in the a section a procedure, procedure was, we actually remove a some of the literals or import or exports something, and wherever the cost decrease is affecting, then if the cost keep it and keep on repeating it. But in a real scenario, we may need to possess the wide variety of moves, and combine them into a, into a sequence of move, that eventually lead to the desired cost.

Other words, we need to consider moves, that may not immediately reduce the cost, this is a actually very difficult problem, that is as I have told you, this is the case, so if you are always thinking that the next solution will be the my best solution. I will keep it then you will, you may get stuck up into a local minima, because you are always coming down but sometimes if you find out that, this solution is something like this here, you may have, may get a better solution sometimes you have to also take the bad solution, and you have to jump up that is what he is saying.

In other words, we it may also need to consider moves, that may not immediately, reduce the cost but that is a more difficult problem to solve, now we are going to see, the simple local search algorithm for logic minimizing the Espresso. So, now what is the previous

part, we have done in the previous part, we have taken a, what do you call this, what you can say the prime implicant form? So, you can say that, we started with this prime implicant form, of a stuff and then we are trying to remove some literal. And something we are trying to go for the next base solution, and so forth.

(Refer Slide Time: 54:53)

A Simple Local Search Algorithm for Logic Minimization

The algorithm has the following basic phases:

1. **EXPAND:** This step expands implicants to their maximum size. Implicants covered by an expanded implicant are removed from further consideration. Quality of result depends on order of implicant expansion. Heuristic methods are used to determine order.
2. **IRREDUNDANT COVER:** Irredundant cover (i.e., no proper subset is also a cover) is extracted from the expanded primes.
3. **REDUCE:** There might exist another cover with fewer terms or fewer literals. Shrink prime implicants to smallest size that still covers the minterms.
4. Repeat the sequence **REDUCE/EXPAND/IRREDUNDANT COVER** to find alternative solutions. Keep doing this as long as new cover improve on best solution.

NPTEL

Handwritten notes in red ink: "1, 2, 3" next to the first three steps, and "1, 2, 3" written vertically at the bottom right.

But now in case of Espresso, as I am taking we are telling from the lecture, that is the main emphasis in this lecture is Espresso here, we will not go about any form of mean terms or merging or something like this. We will rather take the sop form directly, and apply the rules, and keep on applying till, we get a solution, of a desired value, that is we will never modify the solution to get the set of prime implicants. Because that is, what is the complex problem, finding out prime implicants of problem, then finding a subset of them is another complex problem.

So, here what we will do? We start off with this sop formula as given as, as to us, and then we keep on modifying it, some in some means like in this case, we modified by removing 1 literal same thing. We can also try to do it, by modifying by removing some literal, adding some literal and so forth, so moving a taking a literal out adding a literal out, this actually searching the neighbors. So, heuristics this heuristics for logic minimization Espresso, has its own way of searching the neighbors space, so that we are going to see, but again emphasizing.

Here, we are not starting with any kind of prime implicants, so it has very basic 4 steps, first step is expand, this step expands the implicants to their maximal size, implications considered a by an expanded implicants are removed, from the consideration. I will tell you what does it mean, quality of whether depends on the order of implicant expansion, heuristic methods are determine the order. So, what it says, this step expands implicants to their maximum size, that is say for example, you have $x y$ and these are function of z , so you can expand it $x y z$ prime plus $x y$. So, actually we try to expand, but we do not expand whole you take 1, 1 literal at a time like $x y z$, whatever 1 and we expand the functions as much as possible.

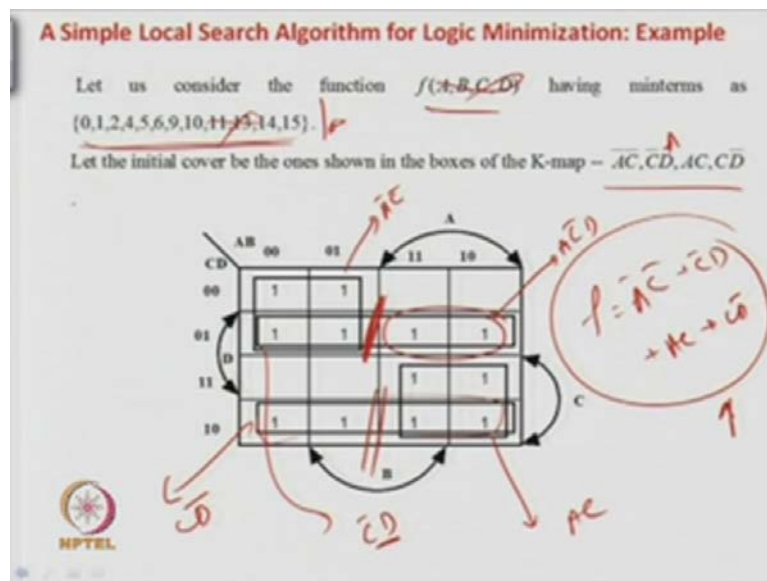
But in canonical representation what you do, you have to expand all the mean terms, in all possible by all possible literals, if you have the function $x y z$, we have to represent all the mean terms, that is you have to expand in all the terms of the $s o p$. In terms of all the literals, but here we select one of them, and we expand fully, then with the maximum size implicants covered by expanded implicants are removed from further consideration. Obviously if you have a implicants all covered by some other, you can eliminate it. Now, the results will depend on the order of implicant expansion, like for example, if you start should, we start with x or y or z , a or b or whatever so that is what again very important in heuristics, is that searching the next solution space. In this case next solution space is you are selecting a literal like $x y z$ or whatever by which you are expanding.

So, if should we start with x , should we start with y or so forth, so depending on which 1 try with next, we will actually give you the how fast you are reaching your solution, that will be determined by it. This is just like the example of a forest and hunter, I have given you which direction you should proceed next will also is very critical in finding on the time criticality of getting to the good solution. So, in this case, there are also lot of algorithms or heuristic algorithms, which will tell you, which based on the previous results, which literal you should start expanding with. Then that is a redundant cover, redundant cover is no proper subset is also a cover is expanded from the expanded set of primes.

So, we will see when come by the expansion, so redundant cover, there is no proper subset is a cover, that is if you have some redundant terms, you can eliminate it reduce. So, in this case what do you do? There might exist another cover with fewer terms or literals, see prime implicants with smallest size series covered the mean terms, so do not

consider this lines. So, we will that we will see, what does that mean? So, here we are not we are not at all bothered primes over here, what we do is that, we this to expand the implicants of the maximum size, then redundant cover, that is if you have some terms which are redundant, it becomes we can eliminate that, and reduce means. Then we exist another cover with the fewer terms or literals that means now we are going to shrink it. So, and then we repeat this procedure, expand redundant and cover to find the alternate, you keep on doing it, till you get the solution, so rather than likely expanding I will give you an example, and again I will come back to this slide.

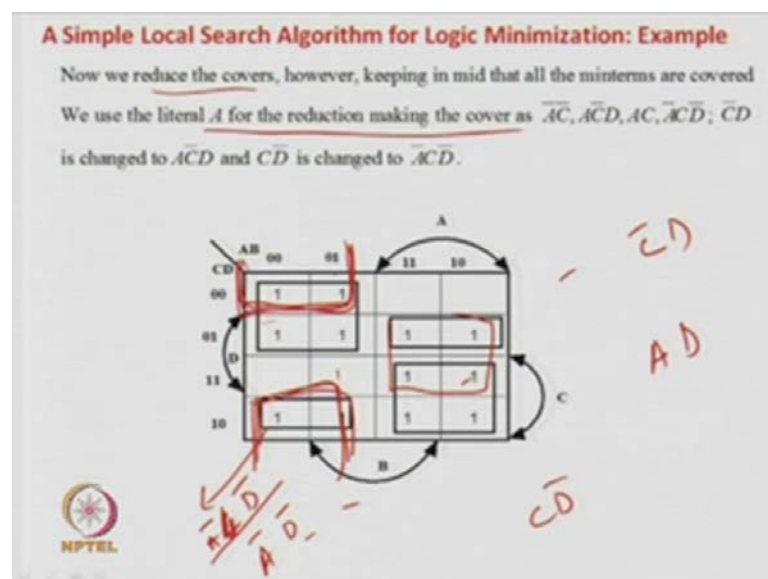
(Refer Slide Time: 58:44)



See this is the function set, so this is the function; this is the these are the mean terms, now this just mean terms are listed, just for example. But Espresso will not be actually involving anything about this, what it will do? See for example, this is the function f is written in terms of A prime C prime plus C prime D plus $A C$ plus something is written over here. I do not bother whether they are mean terms, whether they, so they are prime implicants or whether they are essential prime implicants. Some form is given to this one, and this actually represents this but explicitly also bother about what it represents, so just see this is, what, is the case. So, if you just look at it, what will be this one? This is nothing but A prime C prime and this box is nothing but what you can say C prime D ; this one will corresponding to $A C$, and this one is corresponding to $C D$ prime correct.

So, this how is represented in a, so Espresso also need not do not think about for it, this one is the only thing matters for it. So, this is just like a, this is just a given term for this one, this is just given to you, just for pictorial representation I am showing the map, again repeating this is what is important to Espresso thing, what are A prime C prime implicants, it bothers nothing about it is given to you. So, again I am this is a, this is a very simplified version of Espresso, if you go for the advance version or the implementation version of Espresso, slight changes or some changes over there. So, as the scope of the codes is limited, so we are just discussing the basic idea of Espresso, this is given to you. Next what did you do? Now, we reduce, so it say reduce, so expand either than cover, some of the steps, so first step is reduced. Then you expand then go for redundant cover, reduce expand irredundant cover. So, this you this one you do about, you now what you to do now you have to see this one is there.

(Refer Slide Time: 01:00:39)



Now, we have to reduce, however keeping in mind that all the mean terms are covered, obviously, the mean terms are covered that means 1, you have to reduce means, we are actually shrinking it. So, in this case you have to shrink, reduce means will be shrinking, that in this case we will to put in some more variable, I tell you what does it mean, what are the reduce means over here reduce means shrinking. So, in this case, we will shrink it, and obviously, we have to also think that always, you have to be sure that the function is equivalent, that is whatever was covered. In the previous version of the solution, should also be mean terms, should be also be covered in this case.

So, now what is this we use a for making the reduction, so A for making the reduction means you are actually, so in this case if you remember the old path, so this was your actually the solution this one your solution. And this was on nothing but C D prime, so you are expanding by a mean, so this was actually nothing but C prime D, so C prime D this was C prime D.

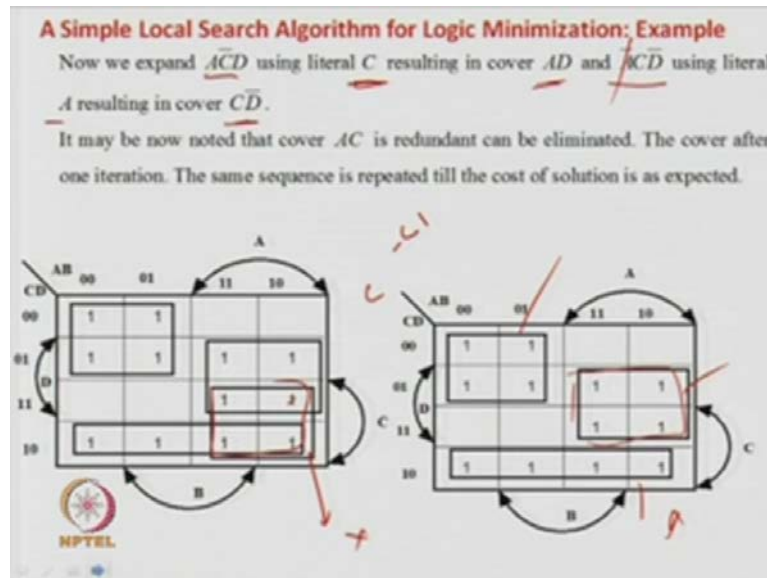
Now, we expand with A that means, what you are reducing, with A that means, you are using A to cut it off. So, in this case, what it will become, you just put it as A, so it will represent is as A C prime, now A CC prime D C this one, initially if you look at the initial solution it is A C prime C prime D A C and C D, so you just break this term. So, that is what is reduce means you are breaking the terms in terms of 1 literal. So, you are putting 1 literal explicitly in this, so in this case you are putting 1 literal A in this case, so in the last example next solution was removing a literal. So, in this case, we are adding a literal, so that is the first step of this one, so you are adding a literal, so it is actually breaking it up.

So, adding a literal means, it will break this up, now only this one will be remain in your term, so it will be nothing like, it will be A. Then it will be C, it will be C prime D C prime D, so in this term we are adding one more term. So, you see the function is now becoming something like this, it is A C prime A C prime D A C, and there was another term A C prime D, there was another term over here A C, that is the case and CC prime D was there. Now, again there was another term, if you have a look at it, so this was another term, which in which A was involved, so this was another term in which A was involved, so that also you have to break it up. So, if you break it up, what is going to happen is, we are going to get this term, so this term is nothing but A prime C D prime, so this is another term, which is coming up initially we had a term something like this. Now, what we are doing, we are selecting the variable a literal A and we are breaking the terms, so there is by, by this term was continue, so this one will get broken, this was another term, which is also get broken.

So, these 2 terms cannot this this terms, this term, cannot be broken, and this term cannot be broken, because they are already inside it. Now, this is what we are getting is the final picture, so now this whole equation this 2; this; this variable and this term, has now change from this one. If you look at it A has been inserted over here, and also in this case some A has been A prime C D prime, so here it is 1 term and so 1 term added over here.

So, this is how we have broken up these terms, so this is called the reduction, so in reduction what you have done, in reduction you select the term literal and you break up this terms. So, in this case these terms are been broken, so it was initially the whole stuff was there you break it up, this was all was there, you break it up, so we have broken this next what you do is?

(Refer Slide Time: 01:04:01)



Expand, so now expansion, what you do you take another variable, and start expanding, so in this case I will tell you, what is there, so let us say another term in which case. So, we expand with literal C, now we are expanding with literal C, so in this case if you see, so in this case was there. So, let us try to expand this, so the expansion will be something like this, and this expansion will be something like this. Now, why is it so now we expand $A C \text{ prime } D$, so this was actually let us look at this, so this if you just have a look at this one, so this one nothing but it was $A C \text{ prime } \text{ and } D$. So, this was, this was this value, this small value so 2 terms where there, we just have a look at this one.

So, now we are expanding in terms of C, and also be careful whether it is possible, in this case it is possible, because this if you expand with C that means, what it will become something like this, and it will be nothing but $A D$. Now, this is possible, because this 2 guys are also a part of the mean, mean term, that means these 2 come also comes in the terms of cover but had it mean the case, that these 2 terms were not there, then if you want to expand by C. Then it will become a invalid solution, in this case the valid

solution therefore, it is possible, now we have expanded in this fashion. Similarly, this also can be expanded by C, now why it can be expanded by C because these 2 are also in in the cover, so it can expand it by C, so what it will become if you extend it by C? It will become C D prime. So, this one is the C D prime, now so this is your final case, now similarly, A C prime D is using the literal, so it is A, so it will become A C D prime, so we are expanding this one and A C D prime using literal A.

So, now you are expanding this one by A, so you are getting this value, now this is your formula now what you can see over here, so there are 2 expansions over here, that is very important. So, this one you have expanded by C, so this one expanded by C this one, now if you say that, this one also I want to expand by C this one, I want to expand by C means, what it will actually become this one. Now, this is what so if you just look at this, so this one was what this one is nothing but your A prime C and D prime something like this. Now, if I want to say that, I want to expand this by C, so by C that means it will become something like this. So, C will be gone, so it will be nothing but A prime will be A prime and so if I make it this way, so it will be A prime C to be A prime, it will be A prime C, if I go it in this way.

So, if I want to you can do this, so if I want to expand it in, so I have done an expansion in this way, so forget about the other ways, which I can, which I want to expand in this way. So, if we want to do it, by this manner, so in this case I see you become, it becomes A D. So, A C prime we are expanding by literal C, so it comes over here, now this; this small term, in this case was this case was A C prime A, this was A prime C and depend. Now, if you want to expand it by C, so it will it will become A prime D prime, so if just want to expand this term by C just eliminating C. So, what it will represent A prime and D prime, so A prime and D prime means, it will be this term, and it will be this 2 term. So, just this 2 term now is actually, A prime C prime, A prime C, and D prime C these 2 term.

Now, if I say that I want to expand it by C, because this one I have expanded by C, so this term was nothing but A C prime D, so just eliminating C prime by expanding by C. I actually go to over this, and this is nothing but your A D, that is possible, because this 2 ones area part of the cover but now and this 2, this 2 stuff are actually A C prime and D. So, if I want to eliminate C from this one that is expand by C, that is remove this, that means it will become A prime and D prime, A prime and D prime are nothing but this

and this so we can also think in this manner. So, in this case your solution will look like something like this, so you will have stuff like this, you will have a stuff like this and but here you can see that, I cannot do anything much over here.

So, this, this term will be there, this term will be there, this term will be there, and this term will be there, so cost will be C, so what this people have done. They have taken another way, so they have said that this I do it by C, that is no problem, so in this case you are actually getting a expansion over this one, that is expanded by C, so you are getting this one you spend by C you get A D. Now, we have seen that if you expand this term by C, you are going to get something like this, and there is no minimization possible. So, now what they are doing the second term is 1, then expanding the terms of literal A, so if you remove literal A, it will become C D C D prime. So, that is you are not expanding it in terms of C, this term again expanding in terms of an A.

So, if you are doing it in this way, so what you are we are getting C D prime, that is nothing but this one, so finally, you are getting this. So, finally, your actually this is the case, which are expansion, now you can see that you are getting a redundant cover, so that is the you know square is redundant, because everything is constant. We can eliminate this, and now you can have this, this and the, and you have gone a got a minimized result, so that is what is Espresso.

So, let me again reiterate what I have done, so what is the first case? First case is reduced, so what do you mean by reduce? Reduce means there may exist another cover with the fewer literals. So, shrink it, so do not think about prime implicants in this case, because the C prime implicants which smallest size that, cover the mean terms. So, in this case that is the resultant, so in this case what is happening, if you keep on reducing, keep on reducing, so finally, we will, have prime implicants in the end, that will actually, cover all mean terms.

That is what is going to be we are going to get it but Espresso does not think about it. It first reduce the, that means, what it will do? It will actually try breaking the terms, if number of literals that is what is actually reduction. So, what it is doing? So, let us just see, so what it has done reduce, so in this case initially, we had the solution was this one. So, initially we had the solution is this one, so what reduce, reduce means you are breaking, so this term you have, have broken out, and this term you have broken out.

And he has chosen A to D so just breaking this one with it will be C D A prime and it will be A C D prime kind of a thing.

So, you are going to get something like this, so that is what is being sold, so reduction the cover as this one is term added, and this one is 1 term added, so this is, this is change to, this is change to something like this, so these are the 2 things, which are generated by reducing the whole thing, by A, now reduce the covers. Now, what you do, now you again go for A expansion, so what is the idea it was reduce expand, so once you have reduced it, now we have to expand, so by expansion can be again in terms of this one. So, in this case so we have they have expanded in terms of there was, if you look at it, so this term they have expanded in terms of C. So, by C, so if you expand it by C, you will get it over here, if you expand this by C, you are going to get something like this but again expanding both by C have not lead us any advantage, because you require 1 2 3 4 terms.

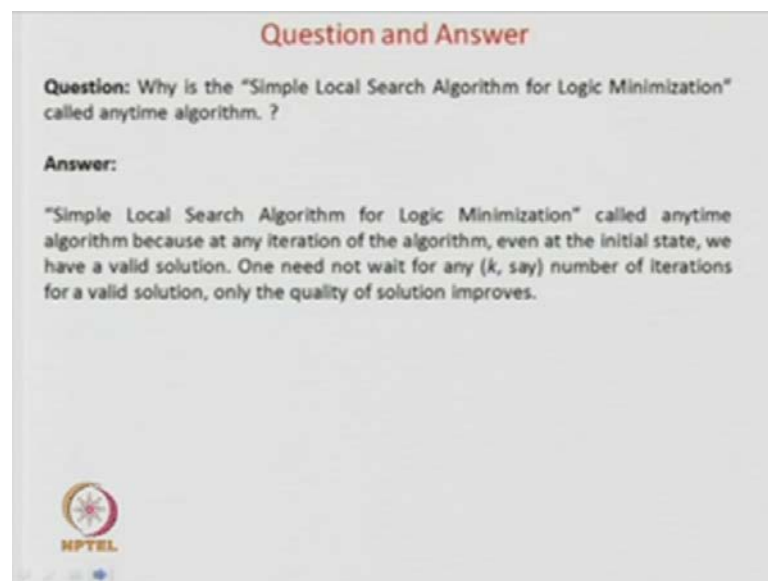
Now, they have again started forgetting that, they have started expanding this one by C 5 but this one by A, if you expand this by A, you are going to get something like this, which is nothing but this term this, this figure. Now, if you do this, then you find out this guy is A, now it becomes redundant, so that is being said. That reduce expand and redundant cover, this they expand expands in, in redundant cover, that is no proper subset is also covered. So, in this case this is your subset, so this becomes A subset this one is your subset, this one also a subset, that means, this is a redundant 1. So, the smaller box this can be eliminated, and you get something like this, so and which is actually now is a is A.

In this case, is a optimal solution but now you have to keep on doing, it you can also try first you try it with breaking up with A, then we have merged with C and A, then again we can start with B, and you keep on doing it till you get the desired solution, of this one that is what is Espresso. Now, again in A what Espresso does, you first even given A any s o p form, you take it as it is, now what you do you start reducing, that means you start breaking of some of the terms, that you are breaking it up. Now, what you do now, you expand, now you expand in A different, different directions, now you again once the expansion is done you finally, redundant cover. If there are, redundant cover eliminated, then your terms have obviously solutions become less and you keep on doing it. So, if I

have to just observe, that this is what exactly is being done, in case of prime minimization as a human being does in case of map.

So, what it will do? He will make some clusters like for example, he may first make a cluster like, he may make the term like this, you may join like this, he may join like, and he may join like this. Then he may think for a while, then he may think for a while saying that, let me try to actually try some other solution. In which case he may say that, this is all solution that is let me try out other way, so he may have broken up this one, that is he is trying to cut some of the things, and expand in another direction, that is cutting in one direction, adding in one direction. So, you just cut this off, and then you start making a club like this, and making a club like this, and then find out this can be eliminated, so Espresso works in a very similar fashion.

(Refer Slide Time: 01:13:22)



As a human being does, so it is first cuts off, some of the terms already present, and then it actually expands in some other directions, and see if something can be eliminated, and it will keep on doing it, till the solution actually reduces. So, this is how is a very in a very broad sense how Espresso works, and how the complexity is reduced. Because we do not have to think about which is the prime implicants, what are the mean terms etcetera, nothing you have to do given some expression form is there, you start you take one alpha. We take one literal and you expand it, now you take another literal, and you, you take, you, you add one literal to all the terms, so that it gets reduced.

Now, you what you do? You take out some other literal, so that it get expanded, and you find out the redundant cover, redundant cover means some of the whole cubes can be removed. If the terms become less it is good, and you keep on repeating it, you get the solution, now this is over all A overall the most widely accepted heuristic in the VLSI design area. We design area, if you ask me what is the best heuristic possible, that has been developed, that is nothing but your Espresso and all logic minimization are standing in the of Espresso. So, the, but these are very, very simple advantage I have told you, expanded version or intelligent version of Espresso what they will do, I mean which is implemented will tell you, which is the first literal to start to reduce, which is the next literal to expand and so forth.

So, choice of the literals are very important told by the algorithm, so this is about the heuristics minimal search, optimality results optimum result. And finally, the algorithm, algorithm Espresso, we saw all the Boolean function minimization, in the modern industry but this was a very mean nature or very small version or simply very very simplified version of heuristic which I put towards you.

Now, I mean let us come to the question answer session, the, the question is why is simple local search algorithm for logic minimization is called anytime algorithm, why or why do I tell you that Espresso is the anytime algorithm. So, what do you mean by any time algorithm, that means end of each solution or end of each, is a solution, that is very another, very important point of heuristic. So, you just run Espresso for one run, so one run what you will do, you will reduce, you will expand go for redundant cover, and a solution is there you can use the solution, again you go for another round, another solution will be there and it is ready.

So, at every deviations some solution is there, they are costly very, very high more down the line you go, the cost will start reducing but at every point of time all the mean terms or whatever solution, you have that is the cubes, will actually cover the function. So, at any point you can stop your heuristic, and you can whatever cubes you have, you they will cover your function that is guaranteed, and that is the solution.

Therefore, it is called the anytime algorithm, that at every iteration, you are going to have a solution. So, that is another very important point of Espresso, that is I mean you never have to let the algorithms stop, for the criteria given a stopping criteria or whatever

you say but at every point, you are going to get a solution, for this one, that is why it is called a anytime algorithm but more time you give the better the solution will be. So, with this we come to this closure of this lecture. In the next lecture, we will see how to minimize sequential circuit, because this was all about combinational circuits. So, next time, we will see, that how do go about minimizing implementation for sequential circuits.

Thank you.